Java15

PROBLEM 1:

**Youngest-oldest 1**

The Pan Am 73 flight from Bombay to New York en route to Karachi and Frankfurt was hijacked by a few Palestinian terrorists at the Karachi International Airport.

The senior flight purser Neerja Banhot had to wither her fear and start evacuating the passengers on board. She pleaded the hijackers to release the oldest and the youngest person in the aircraft. Heeding to her plea the chief of the hijacker agreed to let go the oldest and the youngest. Given the ages of the passengers find the oldest and the youngest.

**Input Format**

The first line of input consists of an integer n, corresponding to the number of passengers in the aircraft.

The next line consists of the age of passengers separated by a space.

**Constraints**

No constrains

**Output Format**

The output prints the youngest and oldest separated by a space.

Print Invalid Input if n or any one of the ages is negative.

**Sample Input 0**

5

18 17 19 12 16

**Sample Output 0**

Youngest=12

Oldest=19

**Sample Input 1**

7

67 23 44 77 24 21 56

**Sample Output 1**

Youngest=21

Oldest=77

CODING :

import java.io.\*;

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner s = new Scanner (System.in);

int n = s.nextInt();

int[] a = new int[n];

for(int i=0;i<n;i++){

a[i]=s.nextInt();

}

int young =a[0];

int older = a[0];

for(int i=1;i<n;i++){

if(a[i]<young){

young = a[i];

}

if(a[i]>older){

older = a[i];

}

}

System.out.println("Youngest="+young);

System.out.println("Oldest="+older);

}

}

Java 15

PROBLEM 2:

**Array 176**

Karthick decided to go to his friend's home. He has reached his friend's home Street. In that place, lots of buildings are there. Then he started to search for his friend's home. He forgot the door number. But he knows the address very well. Finally, he found his friend's home. He wants to know the door number of his friend's home address. Could you please tell me how did he find the door number of his friend's home?

**Input Format**

Input consists of number of buildings in that street and Door numbers for the each home. Karthick friend's home address.

**Constraints**

No constraints

**Output Format**

If he found the home addrress after that returns the door number. If he not found the home,Return the value -1.

**Sample Input 0**

5

7 9 5 3 2

5

**Sample Output 0**

Door Number is 002-DN

**Sample Input 1**

4

8 9 6 4

0

**Sample Output 1**

-1

**Sample Input 2**

4

5 6 7 8

5

**Sample Output 2**

Door Number is 000-DN

CODING :

import java.io.\*;

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner s = new Scanner(System.in);

int n = s.nextInt();

int[] a = new int[n];

for (int i = 0; i < n; i++) {

a[i] = s.nextInt();

}

int search = s.nextInt();

boolean found = false;

for (int i = 0; i < n; i++) {

if (a[i] == search) {

System.out.printf("Door Number is 00"+i+"-DN");

found = true;

break;

}

}

if (!found) {

System.out.println("-1");

}

}

}

Java 15

PROBLEM 3:

**Difference of the character 1**

Given a string S(input consisting) of '' and '#'. The length of the string is variable. The task is to find the minimum number of '' or '#' to make it a valid string. The string is considered valid if the number of '' and '#' are equal. The '' and '#' can be at any position in the string.

Note : The output will be a positive or negative integer based on number of '\*' and '#' in the input string.

(*>#): positive integer (#>*): negative integer (#=\*): 0 Example 1:

Input 1:

*\** -> Value of S

Output :

0 → number of \* and # are equal

**Input Format**

Input consist of one string

**Constraints**

No constraints

**Output Format**

Execute the given output format.check the output data whether it is odd or even.

**Sample Input 0**

#\*\*#\*\*

**Sample Output 0**

The Difference of the character in the given string: 02

**Sample Input 1**

\*\*##\*\*#

**Sample Output 1**

The Difference of the character in the given string: 001

**Sample Input 2**

##\*#\*\*

**Sample Output 2**

0

CODING :

import java.io.\*;

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

String s = sc.next();

int star = 0, hash = 0;

for (char c : s.toCharArray()) {

if (c == '\*') star++;

else if (c == '#') hash++;

}

int diff = star - hash;

if (diff == 0) {

System.out.println(0);

} else {

int absDiff = Math.abs(diff);

String result;

if (absDiff == 1) result = "001";

else if (absDiff < 10) result = "0" + absDiff;

else result = String.valueOf(absDiff);

System.out.println("The Difference of the character in the given string: " + result);

}

}

}

Java 15

PROBLEM 4:

**Count the Positive and Negative Integer Number 1**

sela wants to know how to find the Count of the Positive and Negative Integer Number in the list.could you please help her to find it and implements the program.

**Input Format**

Input Consists of integer list values

**Constraints**

No constraints

**Output Format**

Print the Number of Positive and Negative integer in the list.

**Sample Input 0**

5

3 4 6 -1 -2

**Sample Output 0**

Count of Positive Integer is 3.00

Count of Negative Integer is 2.00

**Sample Input 1**

4

2 7 3 9

**Sample Output 1**

Count of Positive Integer is 4.00

Count of Negative Integer is 0.00

CODING :

import java.io.\*;

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner s = new Scanner (System.in);

int n = s.nextInt();

int[] a = new int[n];

int l=0,m=0;

for(int i=0;i<n;i++){

a[i] = s.nextInt();

if(a[i]>0){

l++;

}

else{

m++;

}

}

System.out.printf("Count of Positive Integer is %.2f\n",(double)l);

System.out.printf("Count of Negative Integer is %.2f\n",(double)m);

}

}

Java 15

PROBLEM 5:

**ARRAY MEAN 4**

Vihaan wants to find the array mean values using programming.Could you please find the array mean value and implements it.

**Input Format**

Input consists of list values.

**Constraints**

No constraints

**Output Format**

Print the array mean values

**Sample Input 0**

5

1 2 3 4 5

**Sample Output 0**

Array Mean Value is 3.00

**Sample Input 1**

3

6 4 3

**Sample Output 1**

Array Mean Value is 4.33

**Sample Input 2**

1

7

**Sample Output 2**

Array Mean Value is 7.00

CODING :

import java.io.\*;

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner s = new Scanner(System.in);

int n = s.nextInt();

int[] a = new int[n];

int sum = 0;

for (int i = 0; i < n; i++) {

a[i] = s.nextInt();

sum += a[i];

}

double mean = (double) sum / n;

System.out.printf("Array Mean Value is %.2f", mean);

}

}

Java 15

PROBLEM 6:

**Count distinct elements 8**

Write a program to count the number of distinct elements in an array.

**Input Format**

Input consists of 1 integer and 1 array. The first integer corresponds to the size of the array.

**Constraints**

No Constraints

**Output Format**

Print the no of the distinct element

**Sample Input 0**

6

2

6

6

4

5

5

**Sample Output 0**

There are 4 distinct elements in the array.

**Sample Input 1**

4

1

1

1

1

**Sample Output 1**

There are 1 distinct element in the array.

CODING :

import java.io.\*;

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

int[] arr = new int[n];

for (int i = 0; i < n; i++) {

arr[i] = sc.nextInt();

}

Set<Integer> distinct = new HashSet<>();

for (int num : arr) {

distinct.add(num);

}

int count = distinct.size();

if (count == 1) {

System.out.println("There are 1 distinct element in the array.");

} else {

System.out.println("There are " + count + " distinct elements in the array.");

}

}

}

Java 15

PROBLEM 7:

**Search an Element 7**

Dhiya wants to search for the presence of an element in an array.could you please help her to search it and implements in program.

**Input Format**

Input consists of one integer and list of array elements. First input corresponds to the array size. Second input corresponds to the array elements. Third input corresponds to finding an array element.

**Constraints**

No constraints

**Output Format**

Print the Statement whether the given elements is Presented in the array or not.

**Sample Input 0**

6

1

2

3

4

5

6

7

**Sample Output 0**

7 is not presented in an array.

**Sample Input 1**

5

2

5

3

6

2

3

**Sample Output 1**

3 is presented in an array.

CODING :

import java.io.\*;

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

int[] arr = new int[n];

for (int i = 0; i < n; i++) {

arr[i] = sc.nextInt();

}

int key = sc.nextInt();

boolean found = false;

for (int i = 0; i < n; i++) {

if (arr[i] == key) {

found = true;

break;

}

}

if (found) {

System.out.println(key + " is presented in an array.");

} else {

System.out.println(key + " is not presented in an array.");

}

}

}

Java 15

PROBLEM 8:

**Find K Largest Elements 1**

Write a Java program to find the k largest elements in a given array. Elements in the array can be in any order.

**Input Format**

First Input corresponds to the array size Second input corresponds to the array elements

**Constraints**

No constraints

**Output Format**

Execute the output in the given format.

**Sample Input 0**

5

7 6 9 3 2

2

**Sample Output 0**

The elements are in the order: 2 3 6 7 9

The Kth value is 2 and Largest elements are 9 7

**Sample Input 1**

6

17 21 12 26 13 31

4

**Sample Output 1**

The elements are in the order: 12 13 17 21 26 31

The Kth value is 4 and Largest elements are 31 26 21 17

CODING :

import java.io.\*;

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

int[] arr = new int[n];

for (int i = 0; i < n; i++) {

arr[i] = sc.nextInt();

}

int k = sc.nextInt();

Arrays.sort(arr);

System.out.print("The elements are in the order: ");

for (int i = 0; i < n; i++) {

System.out.print(arr[i]);

if (i != n - 1) System.out.print(" ");

}

System.out.println();

System.out.print("The Kth value is " + k + " and Largest elements are ");

for (int i = n - 1; i >= n - k; i--) {

System.out.print(arr[i]);

if (i != n - k) System.out.print(" ");

}

}

}

Java 15

PROBLEM 9:

**Sum of positive square 1**

Write a program to find the sum of positive square elements in an array.

**Input Format**

Input consists of 1 integer and 1 array. Integer corresponds to the size of the array.

**Constraints**

No Constraints

**Output Format**

Print the sum of positive square elements in an array.

**Sample Input 0**

3

2

4

6

**Sample Output 0**

56

**Sample Input 1**

5

6

4

2

-5

3

**Sample Output 1**

65

**Sample Input 2**

4

-2

-4

-3

-7

**Sample Output 2**

0

CODING :

import java.io.\*;

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

int[] arr = new int[n];

int sum = 0;

for (int i = 0; i < n; i++) {

arr[i] = sc.nextInt();

if (arr[i] > 0) {

sum += arr[i] \* arr[i];

}

}

System.out.println(sum);

}

}

Java 15

PROBLEM 10:

**Second Array Max And Min 1**

The function accepts an integers arr of size ’length’ as its arguments you are required to return the sum of second largest element from the even positions and second smallest from the odd position of given ‘arr’

Assumption:

All array elements are unique Treat the 0th position as even

**Input Format**

Input

arr:3 2 1 7 5 4

Output

7

**Constraints**

No constraints

**Output Format**

Second largest among even position elements(1 3 5) is 3 Second smallest among odd position element is 4 Thus output is 3+4 = 7

**Sample Input 0**

7

1 8 0 2 3 5 6

**Sample Output 0**

Sum=8

CODING :

import java.io.\*;

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner s = new Scanner(System.in);

int n = s.nextInt();

int[] a = new int[n];

for (int i = 0; i < n; i++) {

a[i] = s.nextInt();

}

List<Integer> evenPos = new ArrayList<>();

List<Integer> oddPos = new ArrayList<>();

for (int i = 0; i < n; i++) {

if (i % 2 == 0) {

evenPos.add(a[i]);

} else {

oddPos.add(a[i]);

}

}

Collections.sort(evenPos, Collections.reverseOrder());

int secondLargestEven = evenPos.get(1);

Collections.sort(oddPos);

int secondSmallestOdd = oddPos.get(1);

int sum = secondLargestEven + secondSmallestOdd;

System.out.println("Sum="+sum);

}

}

Java 15

PROBLEM 11:

**Richest Customer Wealth 7**

You are given an m x n integer grid accounts where accounts[i][j] is the amount of money the ith customer has in the jth bank. Return the wealth that the richest customer has.

A customer's wealth is the amount of money they have in all their bank accounts. The richest customer is the customer that has the maximum wealth.

**Input Format**

Example 1:

Input: accounts = [[1,2,3],[3,2,1]] Output: 6 Explanation: 1st customer has wealth = 1 + 2 + 3 = 6 2nd customer has wealth = 3 + 2 + 1 = 6 Both customers are considered the richest with a wealth of 6 each, so return 6.

Example 2:

Input: accounts = [[1,5],[7,3],[3,5]] Output: 10 Explanation: 1st customer has wealth = 6 2nd customer has wealth = 10 3rd customer has wealth = 8 The 2nd customer is the richest with a wealth of 10.

Example 3:

Input: accounts = [[2,8,7],[7,1,3],[1,9,5]] Output: 17

**Constraints**

m == accounts.length n == accounts[i].length 1 <= m, n <= 50 1 <= accounts[i][j] <= 100

**Output Format**

--

**Sample Input 0**

3 4

1 3 5 2

4 2 6 2

1 4 3 7

**Sample Output 0**

Richest Customer Wealth is 15

CODING :

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int m = sc.nextInt(); // number of customers (rows)

int n = sc.nextInt(); // number of banks (columns)

int[][] accounts = new int[m][n];

for (int i = 0; i < m; i++) {

for (int j = 0; j < n; j++) {

accounts[i][j] = sc.nextInt();

}

}

int maxWealth = 0;

for (int i = 0; i < m; i++) {

int wealth = 0;

for (int j = 0; j < n; j++) {

wealth += accounts[i][j];

}

if (wealth > maxWealth) {

maxWealth = wealth;

}

}

System.out.println("Richest Customer Wealth is " + maxWealth);

}

}

Java 15

PROBLEM 12:

Write a program to execute the transpose of the given 2D matrix.

**Input Format**

first input consists of array size(row and column) second input consists of array elements.

**Constraints**

No Constraints

**Output Format**

Execute the transpose of the given matrix

**Sample Input 0**

3

1

2

3

4

5

6

7

8

9

**Sample Output 0**

Array elements are:

1 2 3

4 5 6

7 8 9

Transpose matrix is:

1 4 7

2 5 8

3 6 9

CODING :

import java.io.\*;

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner s = new Scanner(System.in);

int n = s.nextInt();

int[][] m = new int[n][n];

for (int i = 0; i < n; i++) {

for (int j = 0; j < n; j++) {

m[i][j] = s.nextInt();

}

}

System.out.println("Array elements are:");

for (int i = 0; i < n; i++) {

for (int j = 0; j < n; j++) {

System.out.print(m[i][j]);

if (j != n - 1) System.out.print(" ");

}

System.out.println();

}

System.out.println("Transpose matrix is:");

for (int i = 0; i < n; i++) {

for (int j = 0; j < n; j++) {

System.out.print(m[j][i]);

if (j != n - 1) System.out.print(" ");

}

System.out.println();

}

}

}

Java 15

PROBLEM 13:

**MAXIMUM OF EACH COLUMN 1**

Write a program to find the element of maximum value in each column.

**Input Format**

The first input corresponds to the number of rows. The second input corresponds to the number of columns. The third input corresponds to the elements of the 2D array.

**Constraints**

No constraints

**Output Format**

print the maximum value of each column.

**Sample Input 0**

3

3

1 2 9

8 3 1

4 1 1

**Sample Output 0**

Maximum value in column 1 is 8

Maximum value in column 2 is 3

Maximum value in column 3 is 9

**Sample Input 1**

3

3

7 8 9

4 5 6

1 2 3

**Sample Output 1**

Maximum value in column 1 is 7

Maximum value in column 2 is 8

Maximum value in column 3 is 9

CODING :

import java.io.\*;

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner s = new Scanner(System.in);

int r = s.nextInt();

int c = s.nextInt();

int[][] m = new int[r][c];

for (int i = 0; i < r; i++) {

for (int j = 0; j < c; j++) {

m[i][j] = s.nextInt();

}

}

for (int j = 0; j < c; j++) {

int ma = m[0][j];

for (int i = 1; i < r; i++) {

if (m[i][j] > ma) {

ma = m[i][j];

}

}

System.out.println("Maximum value in column " + (j + 1) + " is " + ma);

}

}

}